REMARKS

This Amendment is in response to the Office Action dated June 1, 2006. Claims 1-20 are pending. Claims 1-20 are rejected. Claims 1, 9 and 15 are amended. Accordingly, claims 1-20 remain pending in the present application.

For the reasons set forth more fully below, Applicant respectfully submits that the present claims are allowable. Consequently, reconsideration, allowance and passage to issue of the present application are respectfully requested.

Applicant petitions to extend the time for a response for one month from September 1, 2006 to October 1, 2006.

Claim Rejections - 35 USC §112

The Examiner states:

Claims 1-20 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the enablement requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to enable one skilled I the art to which I pertains, or with which it is most nearly connected, to make and/or use the invention.

Regarding independent claim 1, embedding a unique identifier included in each of the font specifications in a document <u>at document creation and</u> modification are not disclosed in the specification as claimed.

At document creation, the specification only discloses that "for each font used in the creation of a document, a font file is opened to initiate the font identification (step 20). It should be appreciated that the reference to a font file includes memory resident and network-based (n the Internet) fonts" (page 5, line 4-6) where "each unique font identification is formed from a single font specification of the value: data pairs" (page 5, lines 17-18). No such embedding is disclosed.

Also, nowhere in the specification discloses said embedding at document modification. There is a case, which can be close to embedding, where the specification discloses "once the font specifications are retrieved or calculated, these font specifications are written into the document (step 56) and the document is saved (step 58)" (page 9, lines 203). However, here, it is clear that the font specifications, not the unique identifiers as claimed, are written into the documents, and this is regarding at document saving, not document modification.

Claims 9 and 15 are rejected under the same issue.

Dependent claims 2-8, 10-14, 16-20 are rejected for fully incorporating the deficiencies of their base claims 1, 9 and 15.

The Examiner has stated:

Regarding independent claim 1, embedding a unique identifier included in each of the font specifications in a document at document creation and modification are not disclosed in the specification as claimed.

At document creation, the specification only discloses that "for each font used in the creation of a document, a font file is opened to initiate the font identification (step 20). It should be appreciated that the reference to a font file includes memory resident and network-based (on the Internet) fonts" (page 5, lines 4-6) where "each unique font identification is formed from a single font specification of the value: data pairs" (page 5, lines 17-18). No such embedding is disclosed.

Also, nowhere in the specification discloses said embedding at document modification. There is a case, which may be close to embedding, where the specification discloses "once the font specifications are retrieved or calculated, these font specifications are written into the document (step 56) and the document is saved (step 58)" (page 9, lines 2-3). However, here, it is clear that the font specifications, not the unique identifiers as claimed, are written into the documents, and this is regarding at document saving, not document modification.

Applicant respectfully disagrees.

Embedding font specifications in the document is identified in the recited invention (page 5, line 3), and in Figure 5 "write font specifications into document" at the point of saving the document. Embedding font specifications to respond to document modification are covered when writing the specifications upon document saving. When the user of the recited invention writes the specifications at document save time, both the new document case and the modified document case are covered.

A font specification consists of the necessary font attributes to uniquely identify a font. In terms of usage, a font specification is the set of font attributes similar to a multipart database key. There is no single unique identifier; all font attributes written into the document are used to uniquely identify a font.

Accordingly, Applicant submits that claims 1-20 are clear and definite.

Present Invention

A method and system for ensuring accurate font matching in exchanged documents is disclosed. The method and system comprise forming a database of font specifications. The font specifications includes at least two attributes of a font. Each of the font specifications includes a unique identifier. The unique identifier includes a set of values corresponding to the font attributes for a particular font. The unique identifiers are embedded in the documents at document creation and modification.

The method and system also includes accessing the database by a viewer when saving and opening documents to ensure usage of proper fonts. The viewer uses the appropriate unique identifier and the font specification look up in the database for accurate font matching. None of the cited references singly or in combination do not teach or suggest the cooperation of the recited invention.

Claim Rejections - 35 USC §103

The Examiner states:

Claims 1-2 are rejected under 35 USC 103(a) as being unpatentable over Moore et al. (US Pat No. 5,528,742, 6/18/96) in view of Gartland (US Pat No. 6,512,531, 1/28/03, filed 4/9/99).

The Examiner states:

Regarding independent claim 1, Moore discloses:
Forming a font list, which is equivalent to a font database, for user to select a font to embed in a document (col 4, line 25 col 5, line 9)
Embedding a unique identifier for a font in a document at document creation and modification (col 3, lines 51-60, col 4, line 25 to col 5, line 50, col 7, lines 4-20, col 8, lines 6-52: the font identifier is embedded in a document as a unique identifier when creating document, saving document, and editing document)
accessing the font database by a viewer when saving and opening documents to ensure usage of proper fonts, the viewer using the appropriate unique font identifier to look up in the font database for accurate font matching (col5, lines 20-55, col 8, lines 6-52, col 9, line 64 to col 10, line 19)

Moore does not disclose:

- **Forming a database of font specifications, the font specifications including at least two attributes of a font, each of the font specifications including a unique identifier**

Garland discloses:

- **Forming a database of font specifications, the font specifications including at least two attributes of a font, each of the font specifications including a unique identifier (figures 3 and 4, col 7, lines 20-31: a font source with font descriptions is a database of font**

specifications where the font specification includes at least two attributes of a font; information identifying a font implies that each font has a unique identifier)

It would have been obvious to one of ordinary skill in the art at the time of the invention was made to have incorporated Gartland into Moore for the following reason: Gartland discloses a font description database where each font description includes at least two font attributes providing the advantage to incorporate into the font list, which includes only the font names, in Moore for a font database with complete font information that help viewers to easily determine a font selection via given characteristics.

Applicant respectfully disagrees.

In response to the obviousness assertion by the reviewer; the present invention works in the case where Moore does not; namely box 906, i.e. "Process Document Using Prior Art Method".

There is no need for the recited invention if one embeds the font in the file format as disclosed in Moore. The reason why vendors do not follow this procedure is that fonts are software with license agreements, and it can be a violation of the font license to freely distribute them. There are two ways to design a file format: (1) one where the fonts used in the document are "embedded" and (2) one in which they are not. Moore teaches embedding the actual fonts used in the document into the document structure itself, which is typical in read only formats like the Adobe PDF file format. The recited invention works with file formats that do not have fonts embedded in them, and is unnecessary when the fonts are embedded in the file format as described by Moore. Specifically as recited in claim 1, the unique identifier "includes" a set of values corresponding to the font attributes of a particular font. This recitation is neither taught or suggested by Moore.

The Examiner asserts that a list of fonts embedded in a document as disclosed by Moore constitutes a font database. Moore clearly points out in the initial diagram that the decision box 905 states "Does Header Section Of Retrieved Document Contain A Pointer To An Embedded Font?". As disclosed in Moore, a list of pointers in a file header are not a database of unique font identifiers, the offsets are nothing more than byte offsets for that specific file. When a unique identifier is described in the recited invention, this unique identifier refers to a global naming convention across multiple vendors which are based on attributes in the file. The pointers in Moore are unique only in the context of the document they are embedded in. Moore discusses correct font identification in the context of a file format; the font identification uniqueness is recited in the context of the operating system. These are entirely different concepts than those described in the recited invention.

As for the database reference by the Examiner, a font database in the recited invention comprises font specifications that describe the fonts (uniquely). A font database does not contain fonts; rather, it contains information about fonts. In fact, there is absolutely no reason to store fonts in a database because the operating system API calls for font activation work off of activating a file on that computer's hard drive. Storing a font in a database would be inefficient, as/because one would have to extract the font from the database and write it to a disk file before activating (as described in Moore).

Moving beyond the file format as described in Moore, it is necessary to understand how the operating system understands fonts. The generally accepted practice in Windows and Macintosh computers is to utilize the PostScript name of the font. Two fonts with the same PostScript name can not be active in computer memory at the same time.

In Column 10, lines 7-19; Figure 10, box 1003 is precisely where Moore will fail/Referring to column 10, lines 7-19; Figure 10, box 1003, Moore discloses an application that has no control over how the operating system identifies fonts in memory. The unique identifier that Moore discusses does not exist. The application that Moore teaches will look at the fonts already active on the remote computer and, if they are active in memory, then Moore teaches that the embedded fonts will not be installed and activated.

For example, if the computer contains two fonts of Helvetica from two different foundries (Helvetica A and Helvetica B) that render and print differently, the document will reflow and not render or print properly when the wrong font/one is used. In this example, Helvetica A was used to create the document, but Helvetica B is in a remote computer's memory. When a user views the documents, User views document and Moore's invention will use the font Helvetica B to view it with since the Helvetica B font /it is active in memory and the operating system can not distinguish between the two because both the Helvetica A and Helvetica B fonts have the same postscript name, "Helvetica". Moore can not distinguish between the two fonts, as well. However, the recited invention can distinguish between the two fonts and it will activate the correct font and allow the document to be displayed in the format that was originally intended.

Accordingly, Moore does not teach proper font usage, nor does it teach embedding a unique identifier during the saving of the document for reasons cited above.

Regarding claim 2, which is dependent on claim 1, Moore discloses obtaining a font list for a document being saved, and determining whether font specifications for each font in the font list exist in the database.

Moore discloses obtaining a list of fonts from the computer at the point in time when the document is created and embedding those fonts in the file format of the document. The recited invention stores a list of font attributes in a database for all fonts on the computer and/or a remote font server which uniquely identify the font and discloses a method of querying that database to determine whether or not the font is available locally or remotely.

Regarding claim 3, which is dependent on claim 2, Moore discloses that when the font specifications do exist, the method further comprises retrieving the font specifications for each font from the database.

The font specifications as described by Moore are fonts that a user has decided to embed in the file format. For fonts not embedded in the document, Moore relies on those fonts being active in computer memory. If they are not, then Moore must resort to choosing an alternative font (not the original one used). In the recited invention the user is not limited to searching the operating system but can search the attributes of all fonts on the local system (the/our database of font attributes) irregardless whether they are active in computer memory. Also, Moore can not distinguish between duplicate fonts (not true duplicates; i.e., fonts/ones that render the exact same way). Moore uses PostScript names (Figure 4 and Figure 5) as font identifiers as is common practice. However different font foundries' vendors can and frequently do issue fonts that differ in subtle ways but which/that have the same PostScript name. PostScript names are ambiguous. These PostScript name duplicate fonts can cause problems if the wrong one is active in memory. Moore and the operating system can not distinguish between these; a system and method in accordance with the recited invention can. Using the wrong PostScript name duplicate font can cause the document to reflow awkwardly and is actually a serious problem in print shops and publishing.

Regarding claim 4, which is dependent on claim 2, Moore discloses that when the font specifications do not exist, the method comprises creating the font specifications for each font.

In the recited invention, creating font specifications means a font is added to a database when it does not exist in the database. When added, the font is analyzed, and the attribute metadata is extracted and stored in the database for later use. Claim 4 discusses creation of these attributes.

Moore discusses whether or not a font is active in memory in the remote computer at document viewing time and if not, then the computer takes the embedded font in the document and loads it. The process in Moore and the process in the recited invention are not related in any way.

Regarding claim 5, which is dependent on claim 4, Moore discloses writing the font specifications into the document, and saving the document (embedding a font file, which includes information regarding the font, to a document and saving said document shows writing the font specifications into the document and saving such document).

The font specifications which Moore writes into the document are necessary for the document to function, and are typical of what any vendor of word processing software would do. Moore and presumably others write the PostScript name into the document in places/locations where the file format needs to display text in that font/the desired font. In contrast, the font attributes that are written into the document are not necessary for the file format to function. They do not control text output. Rather, in the recited invention, their purpose is to allow the computer to uniquely identify fonts. Since the file format is not controlled, care must be exercised regarding what is placed into the file.

Regarding claim 6, which is dependent on claim 1, Moore discloses that accessing the database when opening documents further comprises retrieving font specifications from a document being opened (retrieving the font identifier from the document being opened, comparing the font identifier against the list of available fonts in an available font list shows accessing the font database). As before mentioned in claim 1, the font list in Moore includes only the font names without more detailed font descriptions as in Gartland. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have combined Gartland into Moore for a font database with complete font information that helps/help viewers to easily determine a font selection via given font characteristics.

First, all document file formats that render text using fonts write into those file formats at least the PostScript name (i.e., some way to signal to the rendering application to display a given font in a certain manner). The disclosure in Moore is hardly unique and it does not refer to the matter currently under discussion. Moore discloses the same method that all word processing vendors utilize with the extra step of embedding the font. The recited invention provides for injecting executable code into the word processor process that retrieves stored font specifications in the document and accesses an external database of font attributes at file open time, in order to activate in the operating system (outside of the word processing document) the exact fonts necessary to render the document. Moore does not disclose this procedure.

Furthermore, only in hindsight would Gartland be combined into Moore. If Moore had solved the problem of allowing a document to be viewed remotely with the fonts that it was created with, then there would be no need to combine the two ideas. Moore does not solve the problem, due to overlooking the fact that embedding documents in a file format and distributing that file to remote computers is frequently considered a violation of the distribution portion of a font license agreement. Since font licensing information is not stored in a font, there is no way of knowing, when fonts are embedded, whether/ when you embed the fonts whether or not it is legal to do so. In Moore, decision 711 "Is Font Type Embeddable?," the answer is frequently "No". "No" results in "Error Processing".

As to Gartland, file formats are frequently proprietary and not often extendable by third parties. The designer of the file format must design for it to be extended via third party development. In addition, the creator of the word processing program would also have to make it possible to explicitly or implicitly allow a third party to inject code into the word processing process space to write out the font specifications into the file format to extend it in a way that would allow unique font identification to resolve the PostScript name ambiguity problem.

Word processing and other software vendors do not frequently open up their file formats for a variety of reasons; the primary reason being that it locks them in to the existing format and makes it difficult for them to change without breaking third party tools that support the format.

Publishing or making public a proprietary word processing file format also makes it easier for a competing vendor to convert a document into their internal format. And lastly, most vendors can not or do not want to support third parties writing into their format as it may cause unexpected behavior or crashes to their program if the work is done poorly.

As for obtaining access to a word processing program's process space and injecting executable code into it either through an established API or through abilities that are available to the operating environment, the only acceptable way to carry out this process in a commercial application is through a supported API from the vendor. Most vendors do not support a plug-in type infrastructure because this type of infrastructure complicates the architecture and makes the application vulnerable to crashing from a poorly written third party plug-in (it is in the same process space). Further third parties frequently don't understand the host environment well enough to be able to cover all edge cases for post processing a file in the manner necessary to implement the invention.

Since the facilities to make this work are specific and not commonly available, and this would require extraordinary work on the part of the word processing vendor to make the infrastructure available necessary to make extending the file format and injecting code into the open and save processes to activate fonts and write font specifications. Most of the problems are caused by the operating systems adopting a font identification standard that is ambiguous in the first place; i.e., PostScript names.

Regarding claim 7, which is dependent on claim 6, Moore discloses searching the database to locate each font specified by the font specifications in the document (retrieving a font corresponding to the font identifier pointed by the indicator from the permanent storage device and process the document using the retrieved font shows that the font is located and specified by the font identifier).

Moore iterates through the list of embedded fonts' PostScript names. The recited invention provides for searching an external database using a set of attributes to uniquely identify a font. The process of composing the lists of fonts is different in practice between the two approaches but the concept is similar.

Regarding claim 8, which is dependent on claim 7, Moore discloses retrieving each font which has been located. Moore discloses retrieving each font located as long as the fonts are either embedded or active in computer memory. Applicants recite finding and activating fonts that are neither embedded nor active in remote computer memory.

Accordingly applicant respectfully submits that claims 1-8 are allowable for the above-identified reasons. Therefore, Applicant respectfully requests reconsiderations and allowance of the claims as now presented.

Claims 9-20 are system and computer readable medium claims which are similar in scope to claims 1-8. Accordingly Applicant submits that these claims are also allowable for the above-identified reasons.

In view of the foregoing, it is submitted that claims 1-20 are allowable over the cited references and are in condition for allowance. Applicant respectfully requests reconsideration of the rejections and objections to the claims, as now presented.

Applicants' attorney believes this application in condition for allowance. Should any unresolved issues remain, Examiner is invited to call Applicants' attorney at the telephone number indicated below.

Respectfully submitted,
SAWYER LAW GROUP LLP

September 27, 2006

[illegible]
Joseph A. Sawyer, Jr., Reg. No. 30,801
Attorney for Applicant(s)
(650) 493-4540